REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Washington Headquarters Service, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington, DC 20503.
PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.

1. REPORT DATE *(DD-MM-YYYY)*
MARCH 2010

2. REPORT TYPE
Technical Paper Postprint

3. DATES COVERED *(From - To)*
April 2008 – January 2009

4. TITLE AND SUBTITLE
SPRUCE: SYSTEMS AND SOFTWARE PRODUCIBILITY COLLABORATION AND EXPERIMENTATION ENVIRONMENT

5a. CONTRACT NUMBER
FA8750-08-C-0064

5b. GRANT NUMBER
N/A

5c. PROGRAM ELEMENT NUMBER
63781D

6. AUTHOR(S)
Patrick Lardieri, Rick Buskens, Srini Srinivasan, William McKeever and Steven Drager

5d. PROJECT NUMBER
SPRU

5e. TASK NUMBER
CE

5f. WORK UNIT NUMBER
08

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Lockheed Martin Corporation
3 Executive Campus
Cherry Hill, NJ 08002-4103

8. PERFORMING ORGANIZATION REPORT NUMBER
N/A

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
AFRL/RITA
525 Brooks Road
Rome NY 13441-4505

10. SPONSOR/MONITOR'S ACRONYM(S)
N/A

11. SPONSORING/MONITORING AGENCY REPORT NUMBER
AFRL-RI-RS-TP-2010-10

12. DISTRIBUTION AVAILABILITY STATEMENT
Approved for public release; distribution unlimited PA# 88ABW-2009-0686 Date Cleared: 24-February-2009

13. SUPPLEMENTARY NOTES
© 2009 ITT Corp. This article was originally printed in the DoD Software Tech News, April 2009, Volume 12; No. 1. One or more of the authors is a U.S. Government employee working within the scope of their Government job; therefore, the U.S.Government is joint owner of the work and has the right to copy, distribute, and use the work. All other rights are reserved by the copyright owner.

14. ABSTRACT
The Systems and Software Producibility Collaboration and Experimentation Environment (SPRUCE) is an open web portal to bring together DoD software developers, users, and software engineering researchers virtually by enabling their collaboration on specifying and solving software producibility challenge problems. SPRUCE is based on the premise that well articulated and technology users and technology providers. Key SPRUCE features are: self-organizing communities of interest (CoI), dynamically evolving challenge problems with accompanying artifacts, and built-in experimentation facilities to reproduce the problems and evaluate solution benchmarks. To participate in SPRUCE, visit www.sprucecommunity.org and request an account.

15. SUBJECT TERMS
Collaboration, Software Engineering, Challenge Problem, Testbed

16. SECURITY CLASSIFICATION OF: a. REPORT	b. ABSTRACT	c. THIS PAGE	17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
U	U	U	UU	9	William McKeever
					19b. TELEPHONE NUMBER (*Include area code*) N/A

POSTPRINT

SPRUCE: Systems and Software Producibility Collaboration and Experimentation Environment

THE PRIMARY OBJECTIVE OF SPRUCE IS TO ADDRESS THE TECHNOLOGY TRANSITION PROBLEM AND BRIDGE THE "VALLEY OF DISAPPOINTMENT"

by Patrick Lardieri, Rick Buskens, Srini Srinivasan, Lockheed Martin Advanced Technology Laboratories, William McKeever and Steven Drager, Air Force Research Laboratory

Executive Summary

The Systems and Software Producibility Collaboration and Experimentation Environment (SPRUCE[1]) is an open web portal to bring together DoD software developers, users, and software engineering researchers virtually by enabling their collaboration on specifying and solving software producibility challenge problems. SPRUCE is based on the premise that well articulated and bounded problems can spark scientific and engineering innovation in software producibility and help to bridge the gap between technology users and technology providers. Key SPRUCE features are: self-organizing communities of interest (CoI), dynamically evolving challenge problems with accompanying artifacts, and built-in experimentation facilities to reproduce the problems and evaluate solution benchmarks. To participate in SPRUCE, visit *www.sprucecommunity.org* and request an account.

Why SPRUCE?

Consider an engineer or architect working on a DoD program, putting out day-to-day fires and still having to consider and deal with a variety of technical problems at depth. Today, they have no easy means to explore if someone has already encountered similar problems and therefore may have some unique insights to offer. SPRUCE enables sharing of problems and insights by giving the engineer or architect a platform to articulate specific problems, using concrete artifacts and repeatable experiments. By posting the challenge problem, associated artifacts and experiments in SPRUCE, the engineer can also help future programs that could face the same problem.

Next, consider a software producibility researcher or graduate student, focused on building exciting technologies and tools. Today, there is no readily accessible repository of 'real-world' challenge problems and data on which they can demonstrate and validate their technologies. Furthermore, such artifacts may give researchers more insight into actual challenge problems, helping to prevent them from making incorrect assumptions. SPRUCE provides access to such a repository and also enables a perspective on how their tools and technologies may ultimately find their way into practice.

SPRUCE's vision is to satisfy this multitude of stakeholder needs by bringing them together through (a) well-defined, in-depth challenge problems and program-representative artifacts; and (b) repeatable benchmarks and experiments that can be readily conducted in an attached experimentation facility.

Figure 1 reflects the current process within the DoD ecosystem for identifying, developing, and transitioning software producibility technology. Government personnel working DoD acquisition programs coordinate with government personnel working research programs to define software producibility problems and research agendas. The problems are then described and written into research programs' Broad Agency Announcements (BAAs) and performers are asked to bid specific development and transition plans for software producibility solutions.

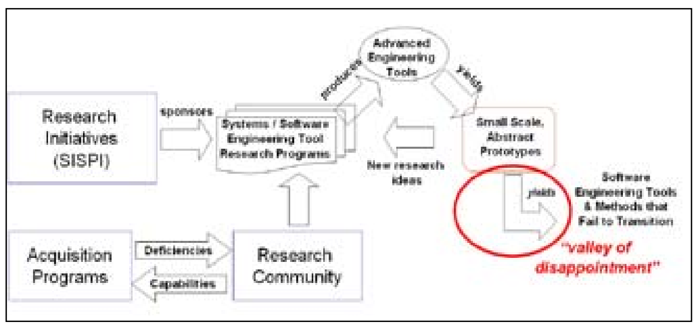


Figure 1: Current technology identification, development, and transition process

Software producibility researchers are then awarded contracts to develop their technology. Unfortunately, these researchers typically have little or no relationship with engineers in the program or domain from which their particular challenge problem is derived. While researchers strive to understand and incorporate deep, specific knowledge about a problem domain, it is hard for them to obtain detailed information and even harder when classification and International Traffic in Arms Regulations (ITAR) issues are involved. Researchers thus have little choice but to design and conduct experiments that are abstract and typically

1

small-scale representations of the real challenge problem. While these results may show the promise of the new technology, they leave a large "credibility gap" in the minds of program engineers about whether the results will transition into the real problem domain. History indicates that it is hard to successfully bridge this gap, leading to the "valley of disappointment" shown in Figure 1. The ultimate success or failure of technology transition thus depends on the ad hoc, opportunistic transition process described above where serendipity of the right people being in the right positions is the primary enabler for success.

The primary objective of SPRUCE is to address the technology transition problem and bridge the "valley of disappointment" described above. SPRUCE emphasizes artifacts (e.g., sanitized DoD application software, computational resources such as specialized avionics processors and workflow management tools and services), typically provided in the context of challenge problems, and experimentation around them to create a common clearinghouse for program engineers and technology researchers to discover joint interests and form collaborations. Collaborations on real world software producibility challenges and the associated experiments using realistic artifacts are the key to successful technology transition.

What is SPRUCE?

SPRUCE is an open, collaborative research and development environment to demonstrate, evaluate, and document the ability of novel tools, methods, techniques, and run-time technologies to yield affordable and more predictable production of software-intensive systems. The key elements of SPRUCE are: (1) a *collaboration environment* that enables and sustains active collaboration between various stakeholders, allowing stakeholders to describe and discuss challenge problems, potential solutions, and experiments to showcase the problems and evaluate solutions, and (2) an *experimentation environment* containing realistic software artifacts and computing systems that promote scientifically rigorous evaluation. The following sections describe the key SPRUCE elements in more detail.

SPRUCE Collaboration Environment

The SPRUCE collaboration environment, implemented as a web portal, seeks to empower its users to define and evolve narrow, well-defined technology problems of mutual interest—but at depth—and seeks to provide them with tools for collaboration and discovery. To achieve this goal, SPRUCE structures its collaboration environment around four basic concepts: communities of interest (CoI), challenge problems, candidate solutions, and experiments and experiment instances (Figure 2).

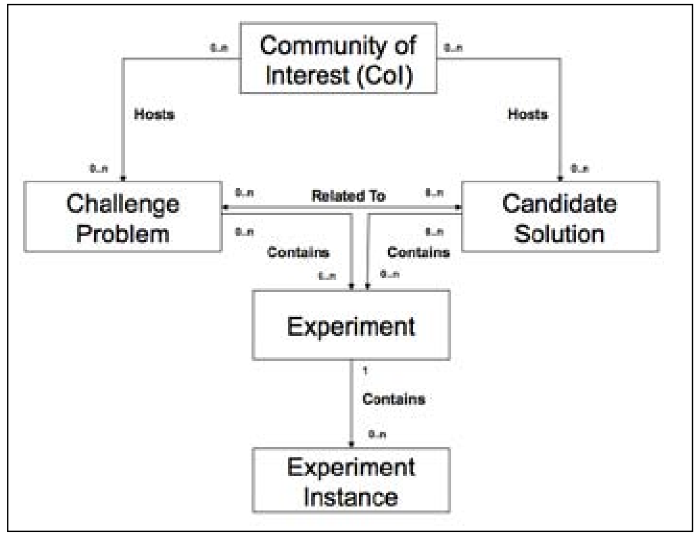


Figure 2: Key SPRUCE Concepts

Communities of Interest (CoI): Communities of interest serve to organize SPRUCE content (i.e., challenge problems, candidate solutions and associated discussions) around broad but focused topic areas. They also serve as a virtual meeting place for SPRUCE users. SPRUCE users can belong to one or more communities of interest.

Challenge Problems: SPRUCE challenge problems represent sanitized versions of realistic problems that may occur on actual DoD acquisition programs. These problems may have occurred on other DoD programs in the past, may express a desire to solve future anticipated problems that would be tedious to solve using existing means, or may provide a context for radically new approaches to systems and software development. As these challenge problems represent a shared concern, they provide an opportunity to bring together the various stakeholders in the DoD software-intensive systems producibility (SISP) ecosystem. SPRUCE encourages and enables DoD programs to submit realistic and sanitized artifacts that accompany challenge problems to attract researchers and provide real-world depth for challenge problems.

Candidate Solutions: SPRUCE candidate solutions describe proposed solutions to SPRUCE challenge problems. Since SPRUCE challenge problems represent realistic problems faced by DoD programs, successful SPRUCE candidate solutions are more amenable to technology transfer. Researchers and tool vendors may, if desired, elect to upload their technology and tools into SPRUCE and to associate licensing conditions with the use of the tools. More likely, however, SPRUCE will be used to highlight specific properties of the tools and solutions and how they address specific challenge problems posed. Researchers and tool vendors can provide links to their solutions for interested

2

SPRUCE users to access.

Experiments: SPRUCE experiments are associated with challenge problems and candidate solutions, and serve two primary purposes: (a) to showcase scenarios described in a challenge problem, so that SPRUCE community members have a repeatable baseline or (b) to evaluate the effectiveness of a particular solution or set of solutions against a benchmark. In the former case, they are best initiated and mediated by the challenge problem provider, whereas a solution provider is best suited to define and conduct the latter kinds of experiments. *Experiment instances* represent an instantiation of a SPRUCE experiment that can be run on actual hardware, including the SPRUCE experimentation environment (discussed in the next section).

As shown in Figure 2, challenge problems, candidate solutions and experiments are interrelated, and each can belong to one or more communities of interest. To facilitate a community's access to collaboration, SPRUCE automatically creates artifact repositories, community wiki and discussion fora (termed 'collaboration facilities') for each of these entities and makes them readily accessible from the entity's main page. The use of social networking tools and instant communication facilities, such as rich text and media chat, as well as member presence information is being considered for future capabilities.

SPRUCE Experimentation Environment

In addition to the web portal, SPRUCE provides an experimentation environment that is available to all SPRUCE users. This environment, comprised of real hardware resources, can be used to illustrate challenge problems and showcase candidate solutions in a repeatable manner on a representative environment. The SPRUCE experimentation environment is based on Emulab (www.emulab.net).

SPRUCE users can access the experimentation environment remotely, request and receive experiment resources, setup desired experiment configurations, download specific operating systems and software, conduct experiments, and collect results. This interaction is done manually with the help of scripts. Results of the experiments can be posted to SPRUCE wiki pages to be shared with other SPRUCE users. Future work will enable the seamless integration of the experimentation environment and the collaboration environment, allowing reuse, cataloging, and automation of many of these functions. The use of a community wiki to post and discuss experimental results enables community-driven peer review and discussion, much like Wikipedia®, thereby enhancing the credibility of the content.

Figure 3 shows the current hardware architecture of the SPRUCE experimentation environment with standard blades and network switches that serve the reconfigurable nature of the experimentation environment. This figure also shows where specialized equipment can be connected and provisioned. SPRUCE will formulate an equipment donation program through which legacy and specialized equipment can be made available for use.

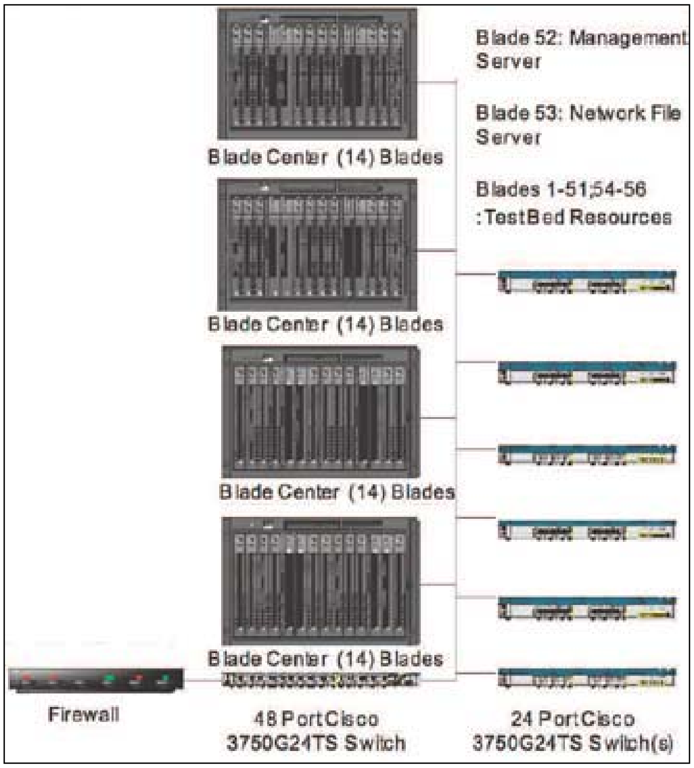


Figure 3: SPRUCE experimentation environment is based on the Emulab technology

Examples in SPRUCE

The SPRUCE portal (http://www.sprucecommunity.org) is currently in Beta phase with a limited number of registered users validating use cases and collaborating around an initial set of challenge problems associated with an initial set of communities of interest. Current SPRUCE communities of interest include: (1) real-time and embedded systems, (2) multi-core architectures, and (3) feature-oriented software analytics. The screen images shown in Figures 4, 5, 6 and 7 showcase initial challenge problems, experiments, and collaborations posted in SPRUCE.

Figure 4 illustrates the main page of a challenge problem that serves as the landing page for the community interested in

3

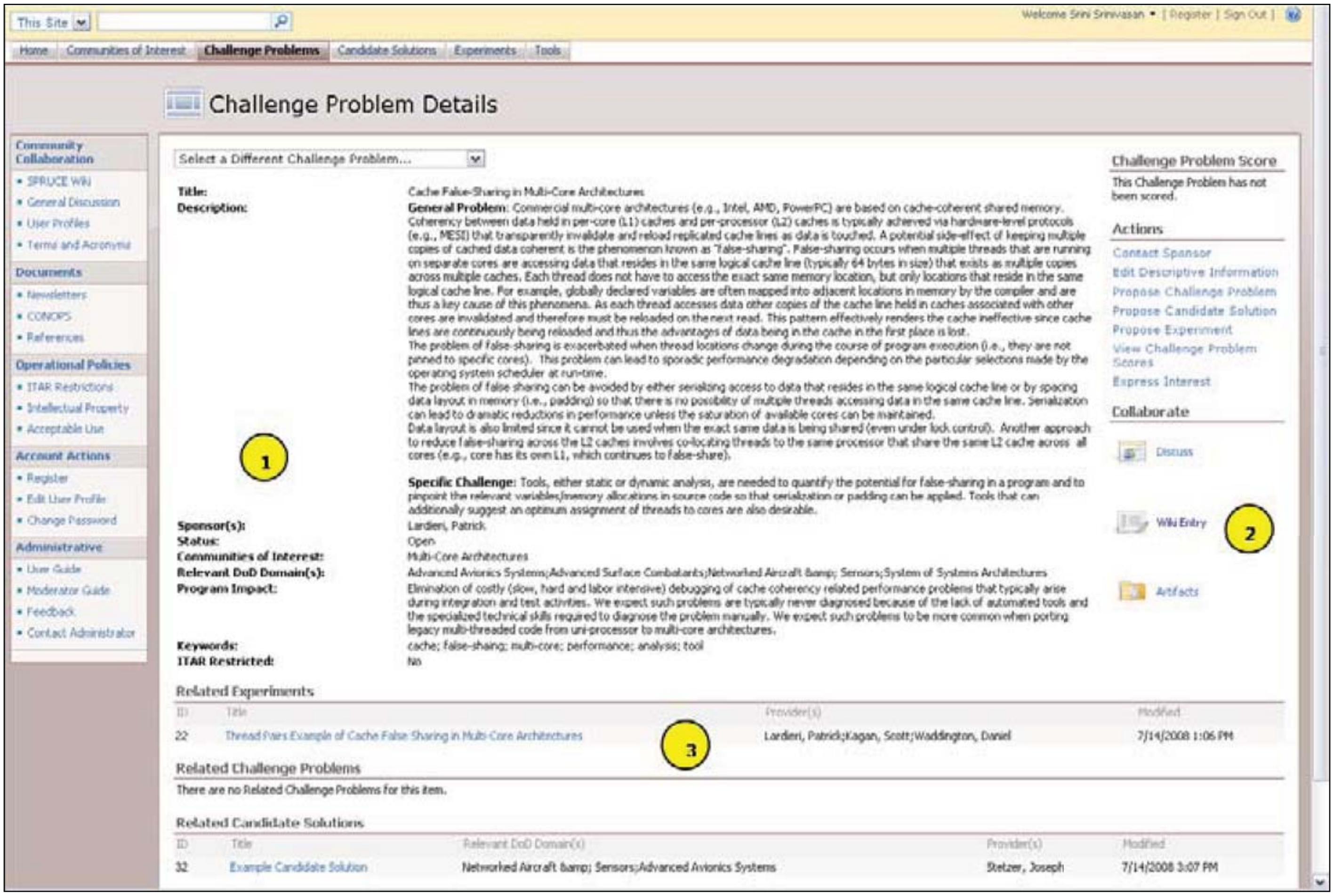


Figure 4: SPRUCE challenge problem main page for the cache false sharing challenge problem

this problem. The main pages of other SPRUCE components are structured similarly. Each challenge problem has associated metadata: title, description, sponsors, keywords and a collection of CoIs (label 1 in Figure 4). There are dedicated collaboration facilities, such as discussion forum topics, wiki pages, and artifact repositories associated with a challenge problem (label 2 in Figure 4). Lists and hotlinks to related SPRUCE entities such as experiments, challenge problems, and candidate solutions are also available for easy navigation and cross-reference(label 3 in Figure 4).

Figure 4 shows a sample challenge problem related to "cache false-sharing" in multi-core architectures. In this problem, conflicting cache requirements of programs running on multiple processor cores constantly invalidate the processor cache and thus cause performance degradation by defeating the purpose for which the cache was designed. This challenge problem is part of the "multi-core architectures" CoI.

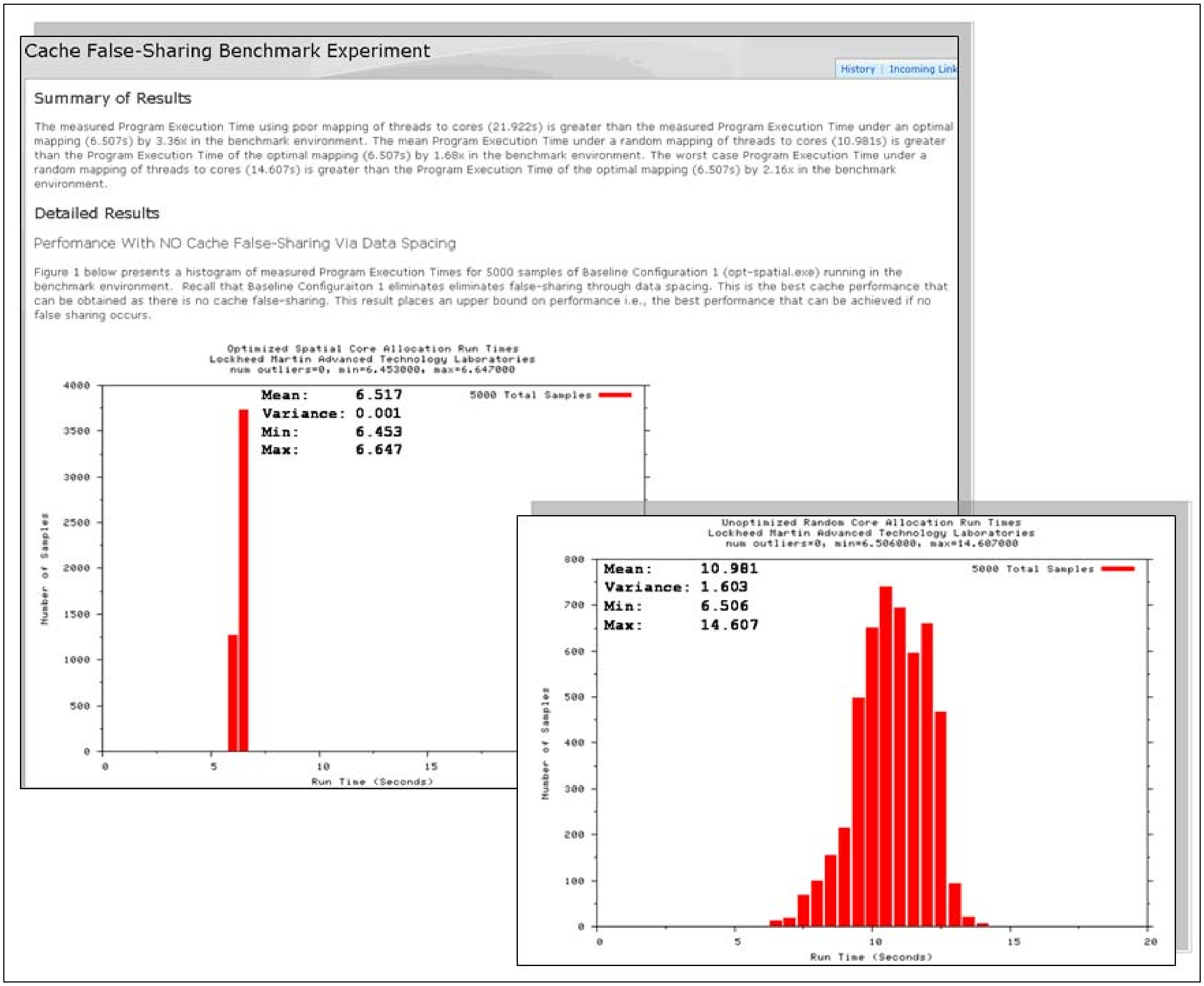


Figure 5: SPRUCE experiment wiki for the “cache false-sharing” problem

Figure 5 shows a wiki entry from an experiment conducted for the “cache false-sharing” problem of Figure 4. The wiki allows for free-form analysis of experiment results and generation of potential ideas for future research. SPRUCE users editing the wiki page can also create links to other places in the SPRUCE portal for easy navigation.

5

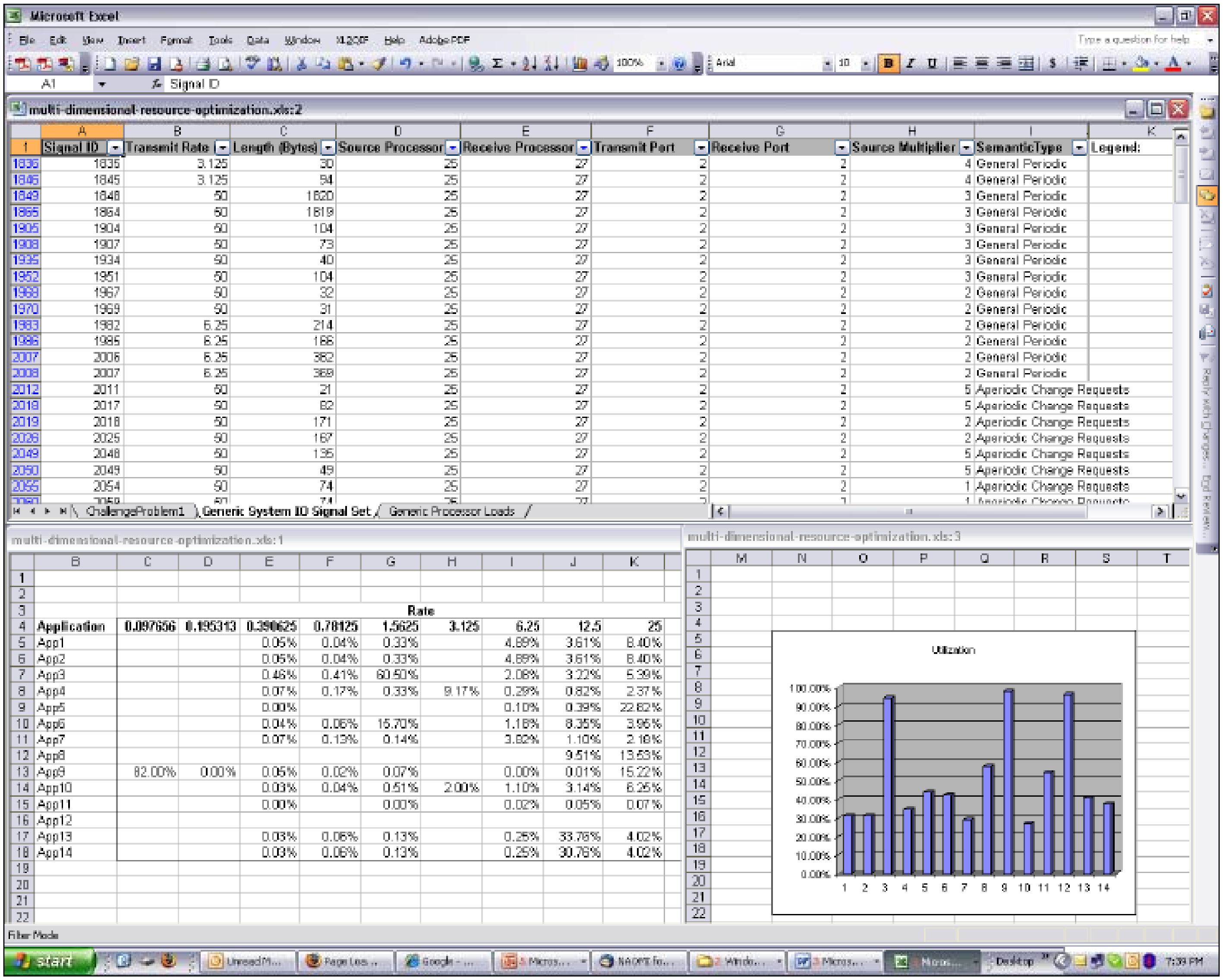


Figure 6: Example artifact for the multi-dimensional resource optimization problem

Figure 6 shows an artifact associated with another challenge problem in multi-dimensional resource optimization. The data set is a sanitized version of an artifact from a military and aeronautics application and represents 11,406 types of network messages flowing across 46 processors. Each message's size, frequency, semantic type, etc., are specified as shown in the top pane. The CPU loading for 14 of the processors, which are available for allocation, are also specified. Collaborations are currently in progress to develop further assumptions, experiments, and solutions to this challenge problem. Since the challenge problem provider also serves as an active moderator for the collaborations, the assumptions, and experiments are guided with an eye toward technology application in the target environment.

Figure 7 shows a screen snapshot from the wiki page of an experiment conducted with a candidate solution (using ASCENT, an algorithm from Vanderbilt University) that was used to analyze a subset of the problem in Figure 6.

--------------------Results---------------------------

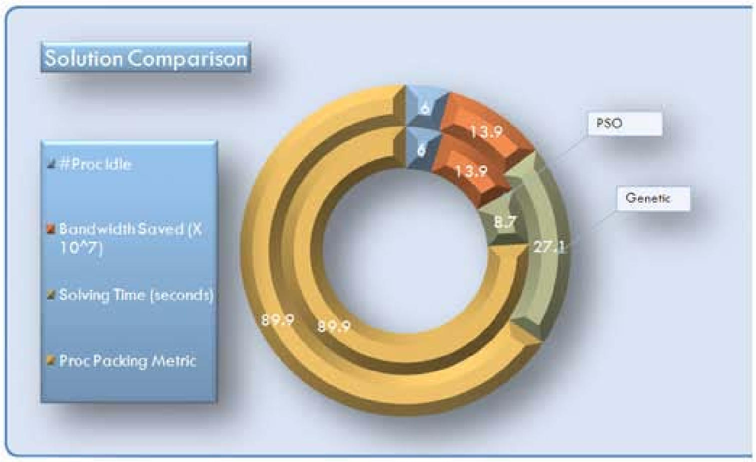
Downloading Results:

The full results evaluation is available in Excel form from: https://www.sprucecommunity.org/Pages/PSO%20Deployment%20Optimization/aero_solution_evaluation.xlsx.

Results Summary:

Empirical Capability Measures	PSO	Genetic
#Proc Idle	6	6
Bandwidth Saved (X 10^7)	13.9	13.9
Solving Time (seconds)	8.7	27.1
Proc Packing Metric	89.9	89.9
Total	118.5	136.9

Capability	PSO	Genetic
Supports Proc Min.	1	1
Supports Custom Bandwidth Calc.	1	1
Supports Startup/Init Scenario	1	1
Supports Harmonic Periods	1	1
Supports Liu & Layland Bounding	1	1
Supports Custom RM Bounding	1	1
Supports Multiple Networks	1	1
Supports Network Gateways	1	1
Supports Co-location Exclusion	1	1
Supports Task Aggregation	1	1
Supports Task Breakup	1	1
Supports Other Non-linear Opt.	1	1
Supports Multipliers	1	1
Supports External Constraint Filters	1	1
Guaranteed Optimal	0	0
Total	14	14



Free Processors:

P11 is free
P12 is free
P13 is free
P2 is free
P26 is free
P28 is free

Deployment Plan:

App1-->P1
App2-->P1

Figure 7: Example screen from a candidate solution documenting the results of an experiment

Enhancing Community Building

Foundational technology to support the discovery of suitable participants in the various SPRUCE communities is currently under development. This technology is based on the notion of affinity relationships between challenge problems, solutions technologies, and personnel interests and publications. The automated collection and construction of affinity relationships will allow SPRUCE to offer query mechanisms such as "since person A is interested in this problem, who else may be interested?", or "given the challenge problems description, construct a list of leading researchers that have published in specified leading journals on related subjects." SPRUCE users can then use the results of such queries to construct invitations to potential collaborators.

Other features under consideration include social networking and community communication mechanisms to facilitate a virtual community, such as instant messaging, presence information, text and media chat.

Getting Involved

If you wish to participate in SPRUCE, please visit the SPRUCE portal (www.sprucecommunity.org) and/or contact the authors. All forms of participation and contributions are welcome – be it through participating with existing communities of interest, establishing and leading new communities of interest, or providing representative DoD artifacts to be shared with the SPRUCE community.

7

Conclusion

Experimentation and collaboration around representative challenge problems in military and aeronautics domains have the potential to bridge the divide between various stakeholders in the DoD ecosystem—a challenge heretofore achieved only through ad hoc and serendipitous engagement between participants. SPRUCE offers an exciting opportunity to address this challenge head on, by providing a platform to promote the desired experimentation and collaboration. The SPRUCE program will undertake a foundational effort in getting an initial set of communities started. Members of the SPRUCE community have the power to sustain and shape the evolution of SPRUCE in the years ahead.

About the Authors

Mr. Patrick Lardieri is a Manager of Advanced Software Technology Research at Lockheed Martin Advanced Technology Laboratories. Mr. Lardieri has over ten years experience in engineering, implementing, designing, testing, demonstrating, and coordinating prototypes in a research and development environment.

Dr. Richard Buskens is a Manager of Advanced Software Technology Research at Lockheed Martin Advanced Technology Laboratories. He has over 20 years of advanced and applied research experience, primarily focused on software producibility activities, and a proven track record for carrying innovative research ideas from concept stage through to product-quality prototypes.

Mr. Srini Srinivasan is a Technology Consultant at Lockheed Martin Advanced Technology Laboratories. Srini has over 20 years of experience in developing and commercializing new technologies.

Mr. Steven Drager is the Advanced Computing Architecture Core Technical Competency lead as well as the technical advisor for the Computing Applications Technology Branch of the Air Force Research Laboratory Information Directorate. Mr. Drager has worked over 20 years at AFRL beginning in reliability physics where he worked wafer-level testing for oxide breakdown, hot carrier degradation, and electromigration.

Mr. William McKeever is a Computer Scientist at the Air Force Research Laboratory Information Directorate in the Computing Applications Technology Branch (AFRL/ RITB). He has worked for AFRL for over five years as a researcher and program manager.

8

Recent Technical Reports from DACS

- **Software Project Management for Software Assurance: A DACS State of the Art Report**

  Released: 9/30/2007

  https://www.thedacs.com/techs/abstracts/abstract.php?dan=347617

- **Enhancing the Development Life Cycle to Produce Secure Software**

  Released: Version 2.0, October 2008

  https://www.thedacs.com/techs/enhanced_life_cycles/

- **Modern Tools to Support DoD Software Intensive System of Systems Cost Estimation: A DACS State of the Art Report**

  Released: August 2007

  https://www.thedacs.com/techs/abstracts/abstract.php?dan=347336

- **A Business Case for Software Process Improvement (2007 Update): Measuring Return on investment from Software Engineering**

  Released: 9/30/2007

  https://www.thedacs.com/techs/abstracts/abstract.php?dan=347616

PDF versions of these reports can be downloaded free of charge by registered DACS visitors who are logged in.

Hard copies can be ordered (for fee) from the DACS store online at **https://store.thedacs.com/**